C Programming Notes

History:

- In 1963 a programming language named CPL (Combined Programming language) was developed at university of Cambridge. The name was chosen because it was combination of all programming language available at that time.
- Problem with CPL was
  - It was too big.
  - And Large number of featured available it made it hard to learn.
- Martin Richards of Cambridge try to solve the problem with CPL by introducing a new programming language named BCPL but it becomes less powerful and to much specific.
- In 1967, Ken Thompson developed a programming language named B but it has also problems similar to that of BCPL.
- By taking all the good points of CPL, BCPL and B Dennis Ritchie developed a programming language named C at AT & T Lab, USA.
- C has achieved lost generality of B and also gained power of CPL.

Structure of C Programs

```
# include <stdio.h>          //header files
#include <conio.h>           //contains information about
int main()
{
    clrscr();          //Clears the Screen
    printf("Wel-Com to C Programming"); // Prints the message in monitor
    getch();           //Holds the screen until we press some key
    return 0;
}
```

Dissection of above program

- Header files stdio.h and conio.h are special files that comes along with C compiler and contains information about library files (Programs).
- stdio.h contains information about printf( ) function that we have used in our program. If we do not use printf stdio.h is not needed to be included.
- Conio.h contains information about clrscr( ) and getch( ) function. If we do not use getch() and clrscr( ) functions in our program conio.h file is not needed to be included.

Another program

```
#include <stdio.h>
#include <conio.h>
int main()
{   clrscr();          //clears the screen
    int a,b,r;          //declares a, b and r as integer variables
    a=2;               //assigns value 2 to the variable a
    b=3;               //assigns value 3 to the variable b
    r=a+b;             // finds the sum of a and b and keeps the result in variable r
    printf("sum=%d",r); // prints the sum in monitor
    getch(); //Holds the screen until we press some key
    return 0;
}
```

C Fundamentals

C character set

Every language has it's own alphabet (character set). Alphabets of C language include:

- Letters: A-Z, a-z
- Digits:0-9
- Special symbols: + - & % # \n ;------- so on

Identifiers

Identifies are names used for various program elements such as names of variables, function names etc.

Rules for naming identifiers:

- ❖ It may contain letters, digits and underscore
- ❖ Must start with underscore or a letter.

Example of valid identifiers:

- ram
- _temp
- result1 etc.

Example of invalid identifiers:

1tax
'a'
order-no
item code

- ➢ underscore is normally used to join identifiers containing two words.
- ➢ C is case Sensitive. This means: if we define two variables named ram and RAM these two are different.

keywords:

Thses are the reserved words having predefined meanings in C. We can not use keywords as programmer-defined identifiers.

Examples of keywords are:

int float case break continue etc.

In standard C language, defined by ANSI, there are 32 keywords. See book for these 32 keywords.

Data Types:

1. **Integer Types**
   - ➢ Used to represent whole numbers.
   - ➢ Keyword int is used to define inter variables as below:
     int a=2;
   - ➢ size of int variables is 2 byte (16 bit).
   - ➢ leftmost bit is used for sign bit.
   - ➢ remaining 15 bits are use for representing magnitude.
   - ➢ range of values represented by int data type is -32768 to +32767

   Formula for calculating the range of value is:
   -2^n to $+2^n-1$
   Where n is the number of bits used to represent magnitude.

-we can use data type qualifiers with data type. Generally used data type qualifiers are short, long, signed and unsigned.
e.g. short int a;

Type	size
int	2 byte
short int	1 byte
long int	4 byte
signed int or int	2 byte
unsigned int	2 byte

2. Character type

- Character types are used to represent single characters.
- Keyword char is used to define character variables as below:
  char c= 'a';
- Characters should be included in single quote. C does not treat r as a character unless we include it in single quote.
- char data type takes size 1 byte.

3. Floating point type.

- Floating point types are used to represent fractional(real) numbers.
- Keyword float is used to define real numbers as below:

float a=2.3f;

- float data type takes the size of 4 byte.
- by using floating point types we can represent a number having six digit precision accurately.
- We can also use keyword double to represent real numbers as below:
  double a=3.4;
- We use the type double when the accuracy provided by float is not sufficient.
- Precision provided by the double data type is 14 digit
- size of double type is 8 byte
- we can use type qualifiers with float and double type as below:
- To further extend the precision we the type long double to represent the real numbers. It takes the size 10 byte.

Constants

- Fixed values that do not change during program execution are called constant.

1. Integer constant

- Any integer valued number are called integer constant.

e.g. 3, -45, +30, 1 etc

- In int a=354;
  here a is a variable but 354 is an integer valued constant.
- long integer constants are represented as below.
  long int a=212L;
  Here a is a variable but 212 is a constant of type long int.

- unsigned integer constants are represented as:
  unsigned int a= 332U;

2. Floating point constants

- Any number that is not a whole number is called floating point constant
  e.g. double d=2.67;
  here d is a variable of type double but 2.67 is a floating point constant.
- constants of type float are represented as:
  float a=3.4f;
- in the above statement, if we omit f 3.4 is considered as constant of type double.

3. Character Constants

- Any single character enclosed in single quote is called character constant.
  e.g. char c='r';
  here c is a variable of type char and 'r' is a character constant.

- C language also has characters that are composed of two or more characters.
  e.g.

\n	new line character.
\t	Tab
\'	single quote.
\"	Double quote
\\	Backward slash.

  So on ..

- Characters that begins with a backslash and is followed by one or more special characters are called escape sequences.

4. String Constant

-A sequence of zero or more characters enclosed in double quote is called string constant.

e.g. "" string constant containing no character.
"a" String constant containing one character.
"nccs" string constant containing more than one character.

Note:- 'a' is a character constant but "a" is a string constant.

Declarations:-

-A declaration associates a group of variables with a specific data type.
-All variables must be declared before it is used in the program.
Example:-

```
int a;          // here variable a is declared to be of type int.
float b,c,d;    //Here variables b, c and d are declared to be of type float.
Char c          //Here variable c is declared to be of type char.
```

Expression:-

- A meaningful combination of variables, constants, operators and function calls is called expression.

e.g.
a+b
a+c-b
a>b;

i = i+1 etc.

Statement:-

- Any expression, declaration, function call or return statement that ends with semi colon is called statement.

e.g.

```
int a,b;
a=b+c;
printf("Hello world:");
 return 0;
```

etc.

#Chapter 2 (Data input and output)

C language is accompanied by a collection of library functions. These Library functions also include large number of input/output functions. In this chapter we read four input/output functions.

-Any identifier followed by opening and closing bracket ((……)) is called function.

Example:

```
int main()   //here main is afunction.
printf("My name is Ankita")  //Here printf(…..) is a function
```

1. getchar function:-

- The getchar() function is used to read a single character from keyboard.
- getchar() function is a part of header file stdio.h .
- To read a character from keyboard using getchar() we write the code as below:

```
char c;
c=getchar();//reads the character from keyboard and put that character
                    in the variable c.
```

2. putchar function:-

- The putchar(character) function is used to print the character specified inside the bracket in monitor.
- putchar(character) function is also a part of header file stdio.h .
- To write a character in a monitor we write the code like below:

```
char a='y';
putchar(a);
```

Code illustrating the use of getchar and putchar

```
#include <stdio.h>
#include <conio.h>
int main(void)
{
  char c;
  clrscr();
  printf("Enter a character\n");
  c=getchar();
  putchar(c);
  getch();
  return 0;
```

}

Dissection of the above program:

directive- #include <stdio.h>

- This is required because we have used library functions printf and getchar in our program and these two library function are part of header file stdio.h.

directive:- #include <conio.h>

- This is required because we have used library functions clrscr() and getch() in our program and these two library function are part of header file conio.h.

Statement:- char c;

- It declares c as a variable of type char.

Statement:- clrscr();

-This is a library function whose information is contained header file conio.h and is used to clear the screen.

Statement:-printf("Enter a character\n");

- This is also a library function whose information is contained header file stdio.h and it prints the message inside the double quote in the monitor.
- \n is a new line character. It takes the cursor in the new line.

Statement: c=getchar();

- getchar reads a character from keyboard and put the character in the variable c.

statement: puchar(c);

- This statement prints the character contained in the variable c in the monitor.

Statement: getch();

- This is used to hold the screen until we press some key.

3. scanf function

- Input data can be entered into the computer from the standard input device by means of the C library function scanf.
- This function can be used to enter any type data(single characters, integers, floating point numbers, string so on) through keyboard.
- Therefore we need to use conversion characters to notify the compiler what type Of data we are going to read as below.

```
int a;
scanf("%d",&a);
//here %d is used to notify the compiler that we are going to read the decimal integer
```

Conversion Character	Meaning
%c	Used to read single character
%d	Used to read decimal integers
%f	Used to read floating point numbers
%s	Used to read strings

So on….. see book for deatail.

For example

-we can read floating point numbers as below:

```
float x;
scanf("%f",&x);
```

-we can also read two or more data item at the same time as below:

```
int a;
```

```
char c;
float x;
scanf("%d %c %f",&a,&c,&x);
```

- here ampersand(&) is called address of operator.

4. printf function:-

- Input data can be displayed in the computers standard output device (monitor) by means of the C library function printf.
- This function can be used to display any type data(single characters, integers, floating point numbers, string so on) in the monitor.
- Therefore we need to use conversion characters to notify the compiler what type Of data we are going to display as shown in above table.

For example

- We can print an integer value as:

```
int a=2;
printf("a=%d",a);
```

- we can print a floating point value as

```
float a=3.2;
printf("a=%f",a);
```

- We can print more than one value as:

```
int a=3;
float b=3.6;
char c='t';
printf("a=%d  b=%f  c=%c",a,b,c);
again consider:
 int a,b,r;
a=4;
b=7;
r=a+b;
printf("sum=%d",r);
```

Example illustrating the use of printf:

```
#include <stdio.h>
#include <conio.h>
int main(void)
{
  int a,b,sum,prod;
  clrscr();
  a=4;
  b=6;
  sum=a+b;
  printf("sum=%d\n",sum);
  prod=a*b;
  printf("Product=%d",prod);
  getch();
  return 0;
}
```

Dissection of the above program

- Here I am only going to explain only the printf statement because all other statement are similar to above program.
- Statement:- printf("Sum=%d\n", sum);
  Here %d is used because we are going to display the value of integer variable sum. Character \n (new line) character is used to take the cursor to the new line.
- statement:- printf("Product=%d",prod);
  Here also %d is used as conversion character because prod is an integer variable.

Example Illustrating the use of scanf

```
#include <stdio.h>
#include <conio.h>
int main(void)
{
/* ----------variable declaration----------*/
     int a,b,diff;
     float c,d,div;
     char x;
/*----------------------------------------------------------------*/
     clrscr();          //Clears the screen
     printf("Enter Two integer values:\n");
     scanf("%d%d",&a,&b);
    //Reads two integers from keyboard, %d %d is used because both variables a nd b
    are of type int.
     diff=a-b;
     printf("difference=%d\n",diff);
     printf("Enter two floating point numbers:\n");
     scanf("%f%f",&c,&d);
    //Reads two floating point numbers.
     div=c/d;
     printf("Division=%f\n",div);
     printf("Enter a character:\n");
     fflush(stdin);
    //Clears the buffer stdin buffer (keyboard buffer);
     scanf("%c",&x);
     putchar(x);
     getch();
     return 0;
}
```

#chapter 3 (Operators and expressions)

- Operator is a symbol that tell the computer to perform some operation on it's operands.

Operators in C language are classified into following categories:

- Arithmetic operators
- Unary operators
- Relational operators
- Logical operators
- Assignment operators
- Conditional operators

1. Arithmetic operators

There are 5 arithmetic operators in C these are:

Operator	purpose
+	Addition
-	Subtraction
*	Multiplication
/	Division
%	Modulus (Remainder after division)

2. Unary Operators

Operators that operates upon single operand are called unary operators**.**

Operator	description	example
-	Negation	a= -b
--	decrement	i--
++	increment	i++

Code Illustrating the use of arithmetic and unary operators:-

```
#include <stdio.h>
#include <conio.h>
int main(void)
{
    int a,b,r;
    clrscr();
    a=10;
    b=3;
    r=a%b; //modulus operator gives remainder of the division.
    printf("Remainder=%d\n",r);
    a++; //Increments the value of a by 1
    printf("After increment a=%d\n",a);
    b--; //decrements the value of b by 1
    printf("After decrement b=%d\n",b);
    getch();
    return 0;
}
```

3. Relational operators:

There are 6 relational operators in C.

Operator	Meaning	Example
<	Less than	a<b
<=	Less than or equal to	a<= b
>	Greater than	a>b
>=	Greater than or equal to	a>=b
= =	Equal to	a = = b
!=	Not equal to	a!=b

- Expression that uses relational operators gives the value either true or false.

e.g

a=3;
b=2;
a>b; (value is true)
a<b; (value is false)
a = = b; (value is false)
a!=b; (value is true)

- Variables that have value only true and false are called Boolean variables.
- Boolean values are actually integer values where any non-zero value (usually 1) is treated as true and zero is treated as false values.

Code Illustrating the use of relational operator

```
#include <stdio.h>
#include <conio.h>
int main(void)
{
    int a,b,r;
    clrscr();
    a=10;
    b=3;
    r=a>b; //since a>b is true value of r becomes 1;
    printf("value=%d\n",r);
    r=(a<=b); //since the condition is false value of r becomes 0;
    printf("value=%d\n",r);
    getch();
    return 0;
}
```

4. Logical operators

- There are three logical operators in C.

Operator	expression	Meaning
&& (and)	expr1 && expr2	Return true if both expressions gives the true value otherwise it returns false.
|| (OR)	expr1 || expr2	Return true if at least one expression gives the true value otherwise it returns false.
! (NOT)	!expr	Returns true if the expr gives false otherwise it returns false

Examples:
a=3;
b=2;
c=5;
(a>b) && (c>a) (Gives true value)
(a>b) && (a>c) (gives false value)
(a>b) || (a>c) (gives true value)
!(a>b) ((gives false value)

- Try to write an example that illustrates the use of logical operators:-

5. Assignment Operator

- Assignment operator is used to assign the value in right side of the operator to the variable in left side of the operator.
- There are several assignment operators in C.
- Commonly used assignment operator is =.

Example: a=4; b=678; a=c*d; etc

- We can also use assignment operator as below:
  a=b=c=45;

- This statement assigns the value 45 to all three variable a, b and c.
- we can also use assignment operator as below.
  a +=1 is equivalent to a = a+1;
  a - =1 is equivalent to a = a-b;
- we use this form of assignment operator with other operators (*,/,%, &&,|| so on) also.

6. Conditional Operator

- It is a ternary operator because it takes three operand.
- the form of conditional operator is exp1 ? exp2 : exp3.

Example:

(a<0) ? 0 :100

Meaning:- if the condition a<0 (exp1) is evaluated to be true value of a becomes zero (exp2) otherwise value of a becomes 100 (exp3).

```
#include <stdio.h>
#include <conio.h>
int main(void)
{
    int a;
    clrscr();
    a=-2;
    a=(a<0)?0:10;
    printf("value=%d\n",a);
    getch();
    return 0;
}
```